LA-UR-17-27350
Approved for public release; distribution is unlimited.

Title: FleCSPH : A New SPH Code Based on FleCSI Framework

Author(s): Lim, Hyun
Loiseau, Julien

Intended for: Exit presentation

Issued: 2017-08-16

**Disclaimer:**
Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the Los Alamos National Security, LLC for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Los Alamos
NATIONAL LABORATORY
EST. 1943

Delivering science and technology
to protect our nation
and promote world stability

National Nuclear Security Administration

Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA

# FleCSPH : A New SPH Code Based on FleCSI Framework

**Hyun Lim[1 2], Julien Loiseau[3 4]**

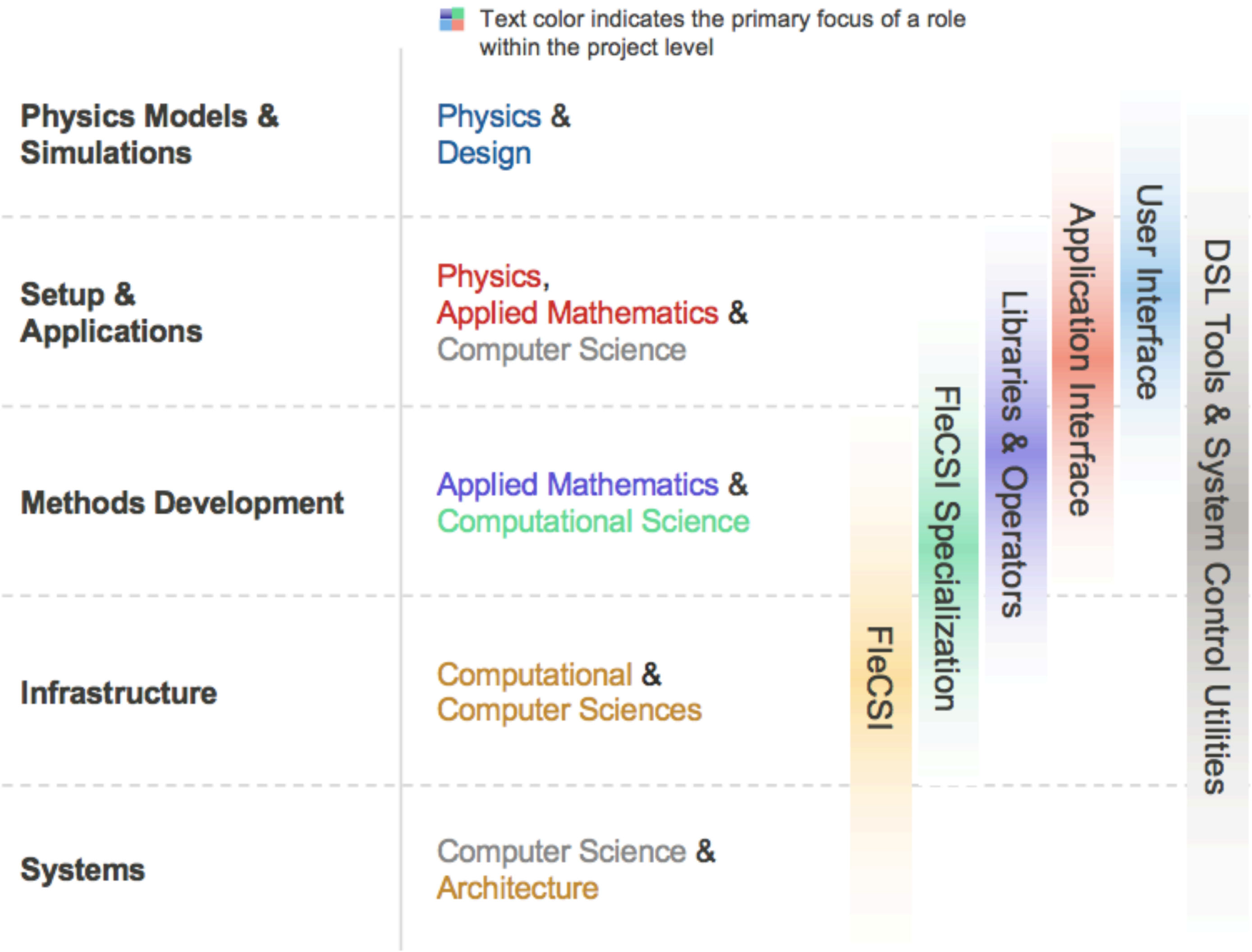
1. **Department of Physics and Astronomy, Brigham Young University**
2. **CCS-2, LANL**
3. **CReSTIC, University of Reims Champagne-Ardenne**
4. **CCS-7, LANL**

Work with Ben Bergen, Nick Moss, Christoph Junghans, Pascal Grosset, Oleg Korobkin, Wes Even, Brett Okhuysen

Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA

# LA-RISTRA project

- **Provide flexible computational infrastructures to solve multi-physics problems**
- **Current main projects involve**
  - FleCSI
  - FleCSALE
  - FleCSPH
  - More in future



Text color indicates the primary focus of a role within the project level

| | |
|---|---|
| Physics Models & Simulations | Physics & Design |
| Setup & Applications | Physics, Applied Mathematics & Computer Science |
| Methods Development | Applied Mathematics & Computational Science |
| Infrastructure | Computational & Computer Sciences |
| Systems | Computer Science & Architecture |

FleCSI

FleCSI Specialization

Libraries & Operators

Application Interface

User Interface

DSL Tools & System Control Utilities

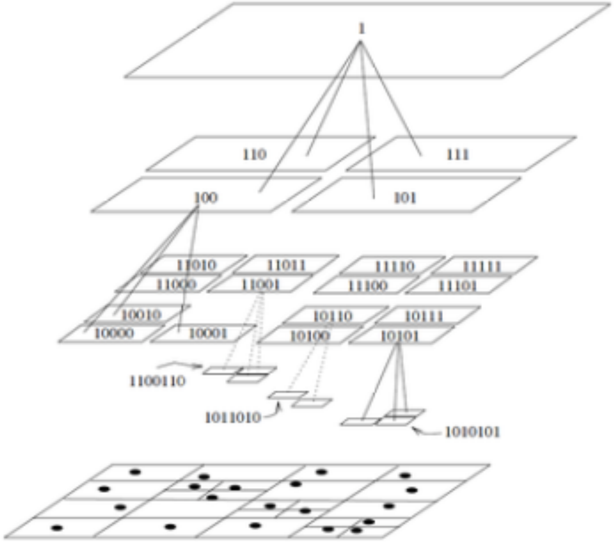
# What is the FleCSI??

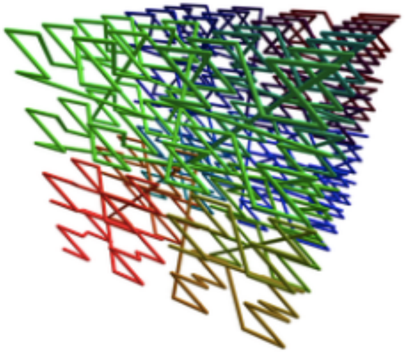
# FleCSI

- **FleCSI is a C++ programming system for developing multi-physics simulation codes**
- **Runtime abstraction layer**
  - High-level user interface, mid-level static specialization etc..
- **Programming model**
  - Control, execution, and data models
- **Useful data structure support**
  - Mesh, N-Tree, and KD-Tree

# FleCSI : Tree Data Structures

- **Tree topology**
  - Support n-tree (also hashed n-tree)
  - Constant-time neighbor look-up
  - Morton ordering
  - Refinement and coarsening
  - Applications: SPH, N-body, AMR, Complex Flows, Monte Carlo, Molecular Dynamics

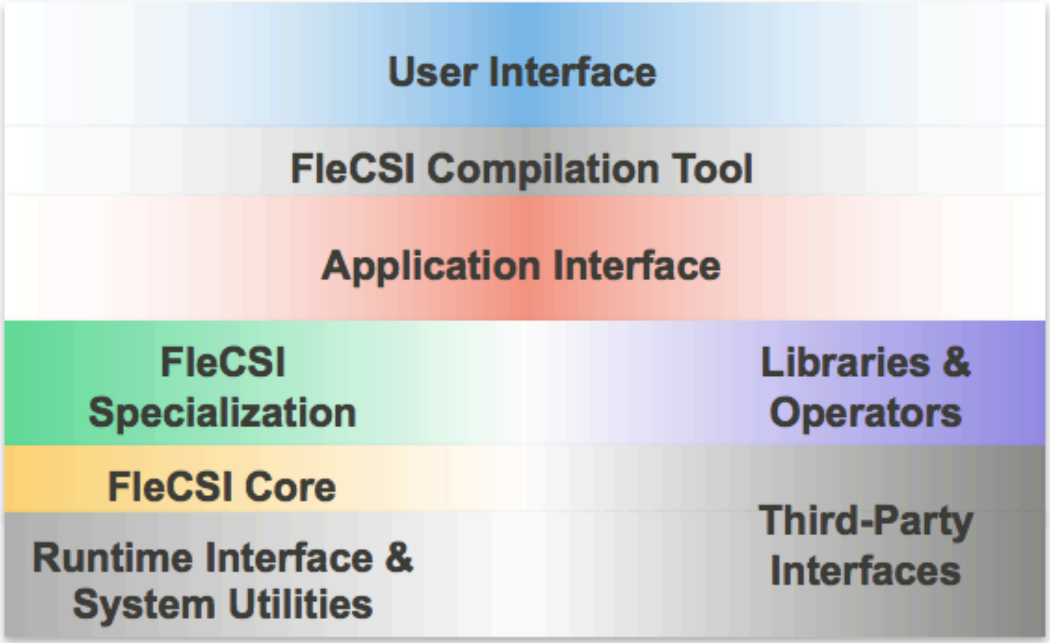




# FleCSI : Where does it sit in the software stack?

Provides high-level, domain-specific
interface that requires syntax
and semantics
not available in C++

Data & Execution Model
Implementations



User Interface

FleCSI Compilation Tool

Application Interface

FleCSI Specialization

Libraries & Operators

FleCSI Core

Runtime Interface & System Utilities

Third-Party Interfaces

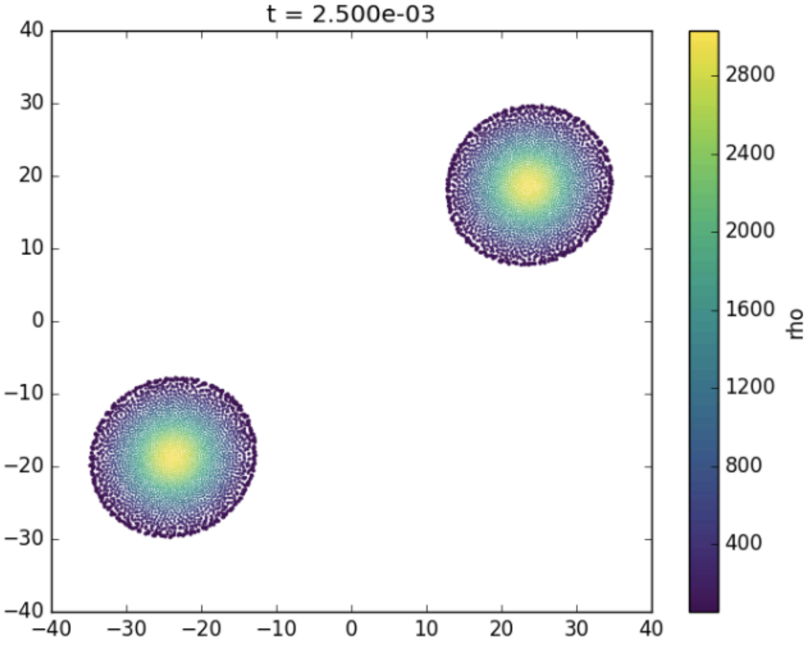
# What is the FleCSPH??

# FleCSPH

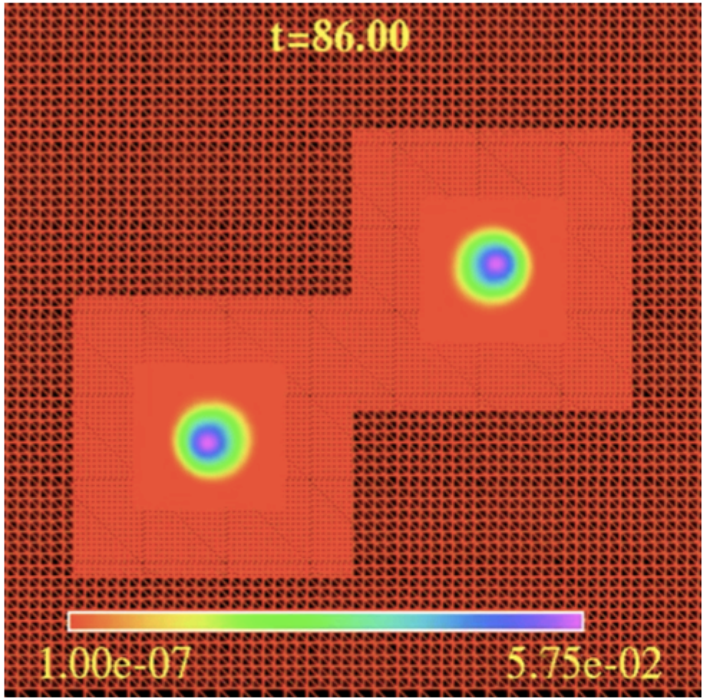
- **FleCSPH is an implementation of smoothed particle hydrodynamics (SPH) based on the FleCSI frame work**
- **Started to develop April 2017**
- **Julien Loiseau and Hyun Lim are main developers**
- **Solves Lagrangian conservation equations for mass, energy and momentum of an ideal fluid with Newtonian gravity**

# Why SPH?

- Exactly conserves mass, linear & angular momentum, and energy
- Perfectly handle vacuum, deformation



- Artificial atmosphere required



# Why SPH?

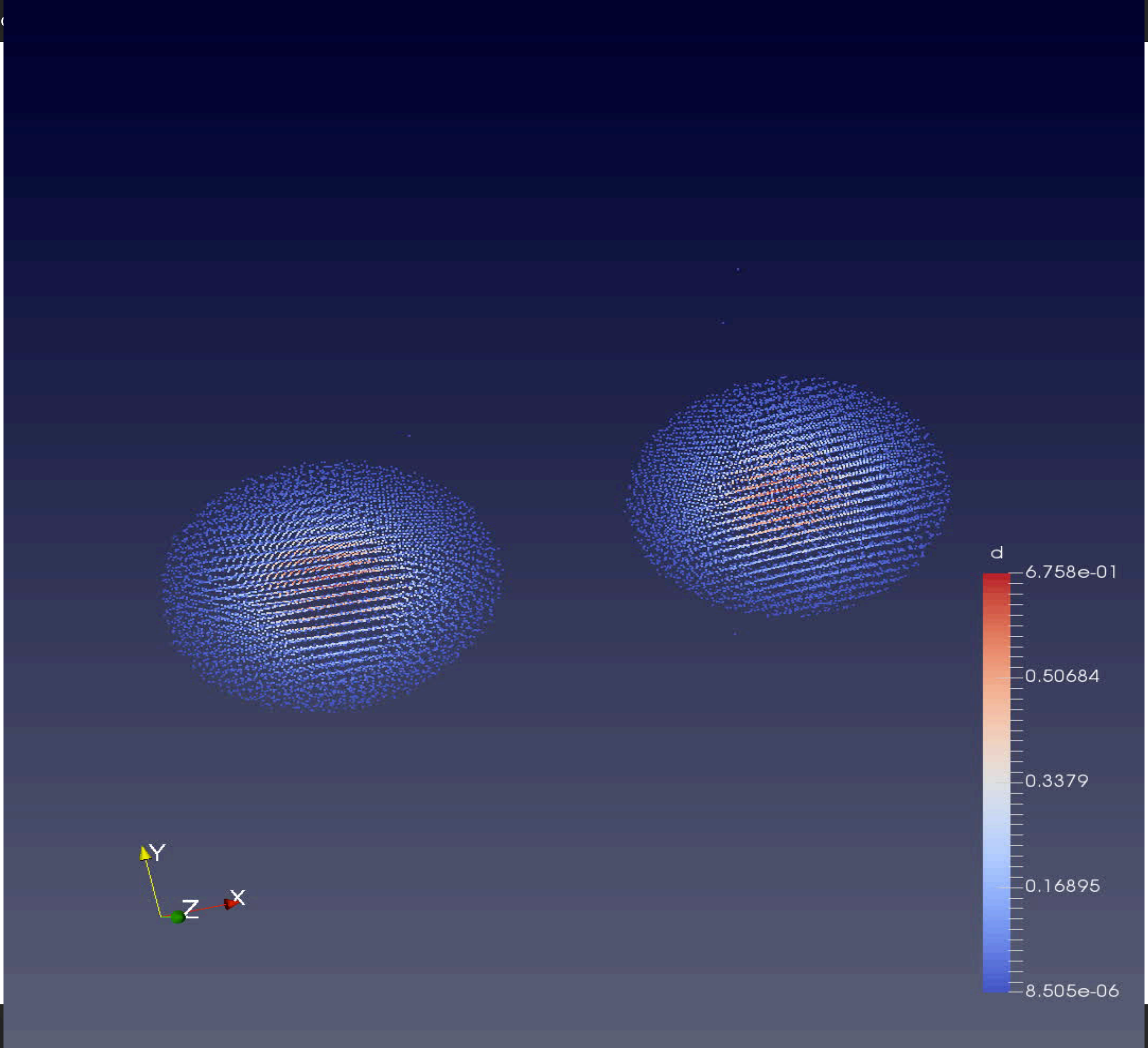
- **SPH is interesting method to test tree data structure**
- **Suitable method for combining with FleCSI infrastructure**
- **Perfect test case for complicated problem for Exascale computing**
- **Has a lot of testable applications:**
  - Space filling curve
  - Neighbor searching
  - Different tree traversal and sorting algorithms

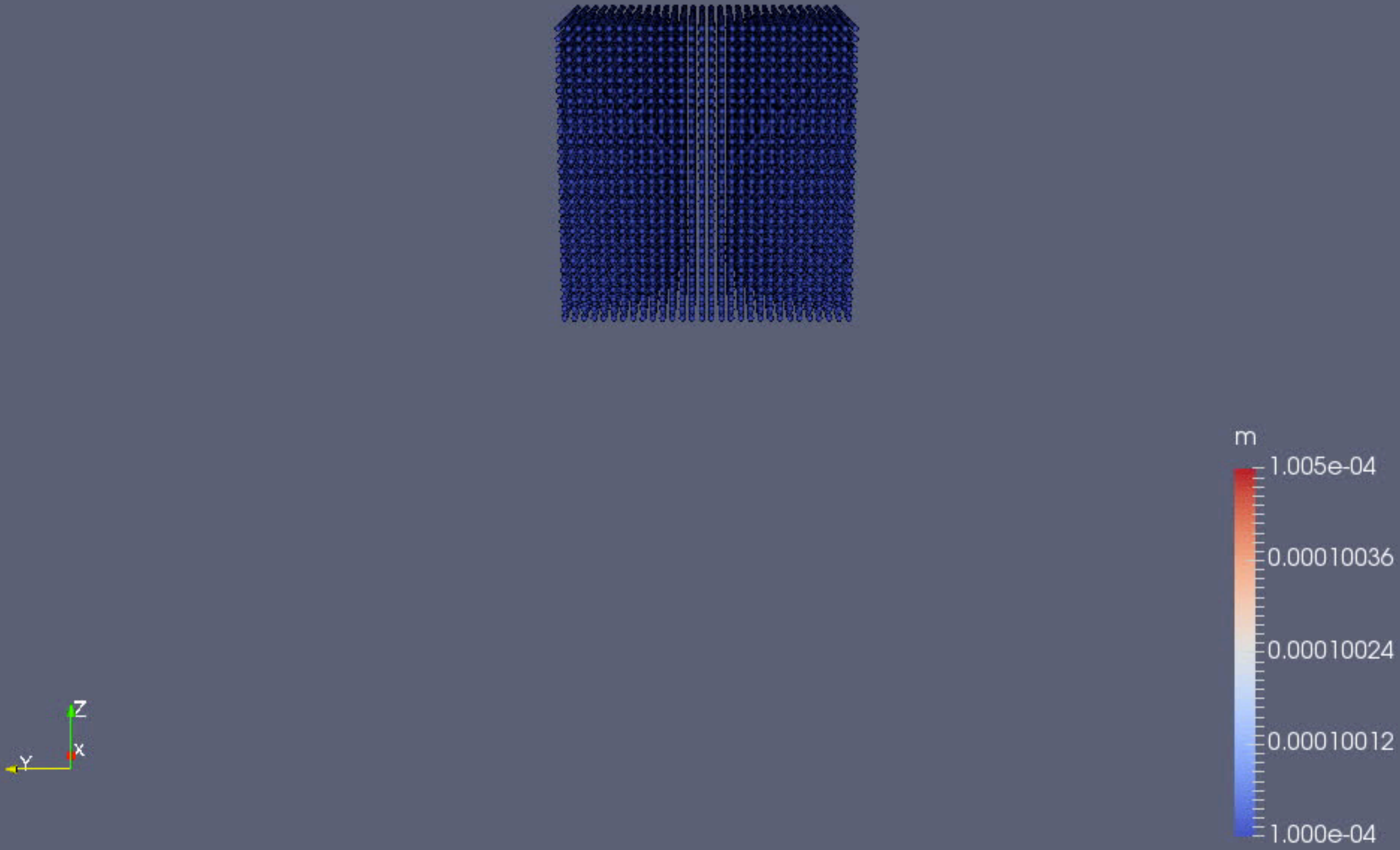
# FleCSPH : Current Development

- **FleCSPH has capabilities:**
  - Cubic spline kernel approximation
  - Hydrodynamics with Newtonian gravity
  - I/O with h5part
  - Parallel and distributed memory shared scheme
  - Parallel jobs are executed through MPI and Legion

# FleCSPH : Preliminary Results

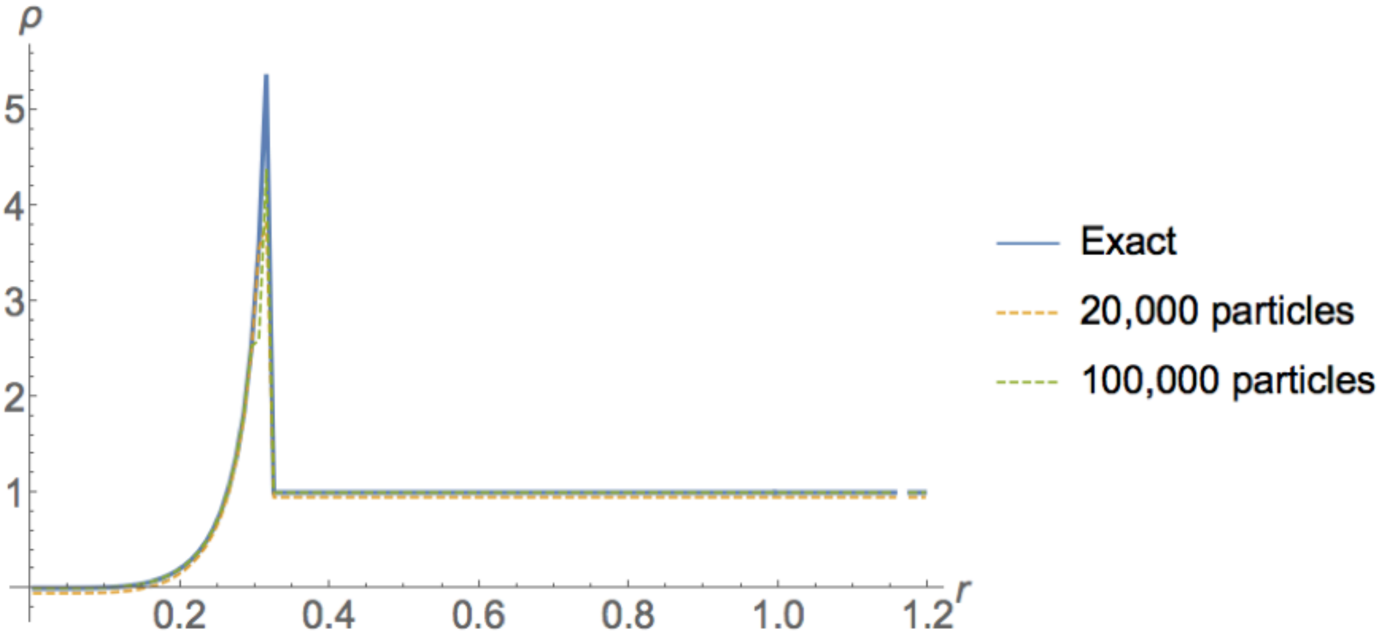
- **Currently, we tested several different problems:**
  - 1D Sod shock tube
  - 2D Sedov blast wave
  - 3D Fluid problem : Water cube drop
  - 3D Compact binary objects
    - Double white dwarfs
    - Binary Neutron Stars
  - Working on more astrophysical problems…


d
6.758e-01
0.50684
0.3379
0.16895
8.505e-06
Y
Z
X

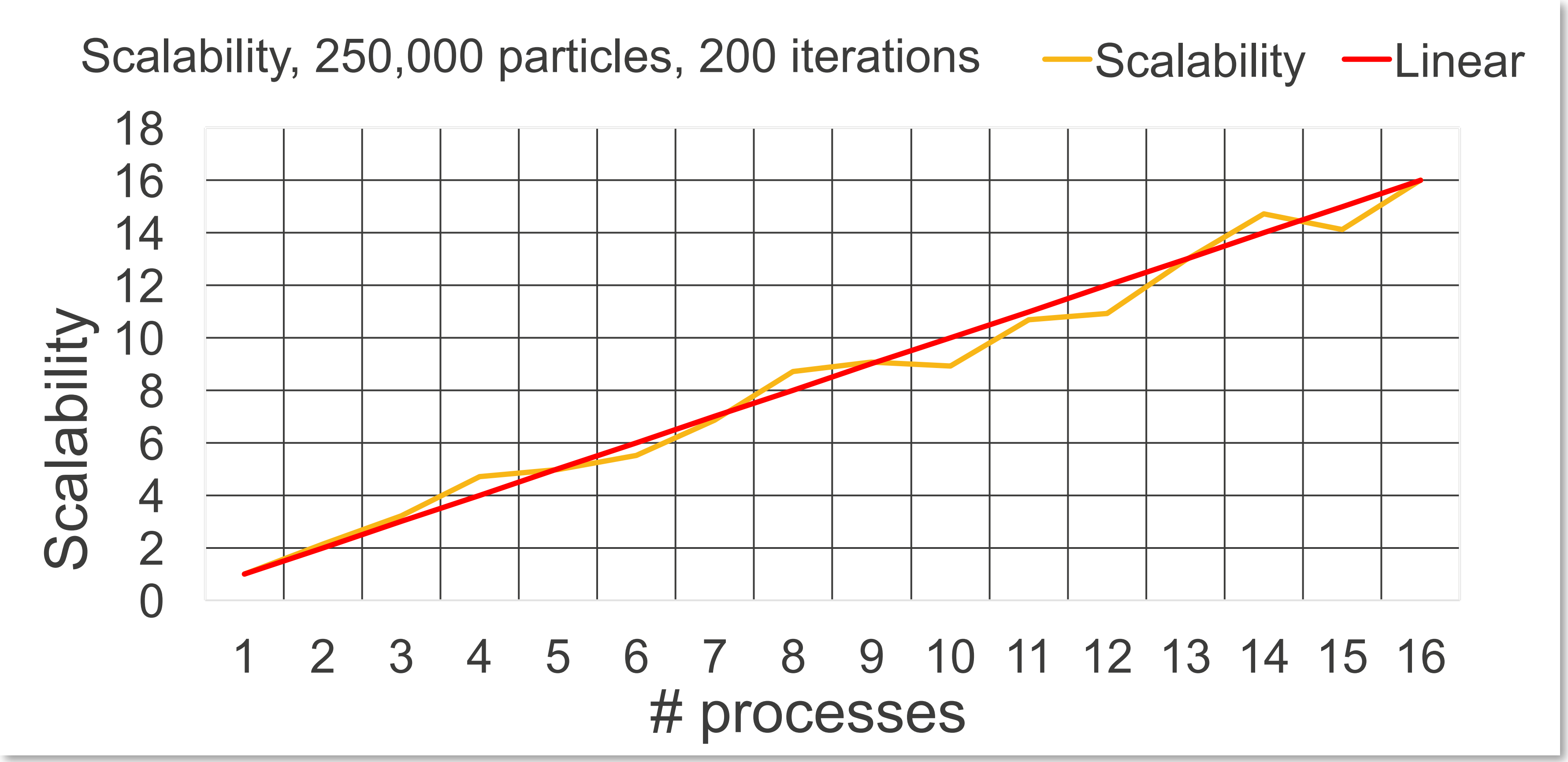

m
1.005e-04
0.00010036
0.00010024
0.00010012
1.000e-04
Z
X
Y

# Performance Results

Density plot for comparison Sedov blast wave result



# Performance Results



# Conclusion and Future Works

- **FleCSPH is still developing!!**
- **Need to put different functionalities:**
  - Different types of EOS (includes tabulated), more microphysics
  - Nuclear network
  - ID generator that is suitable FleCSPH
  - Different smoothing kernels
  - Add gravitational wave radiation reaction
  - And more…

# Conclusion and Future Works

- **Add complete framework into FleCSI:**
  - Add different space filling curve : Hilbert, Peano, Gosper
  - More task-based runtime model
  - More optimized and multi-purpose tree code

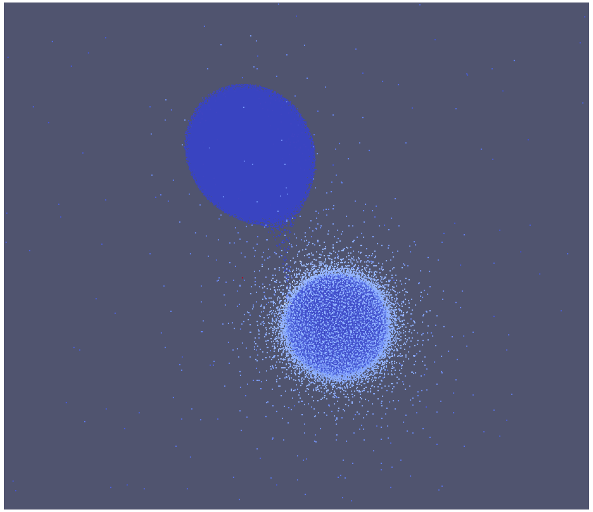
# To use FleCSPH.....

- Go to our github page

https://github.com/laristra/flecsph

- Follow the instruction and enjoy!!

- If you have any questions, please contact Hyun Lim and Julien Loiseau

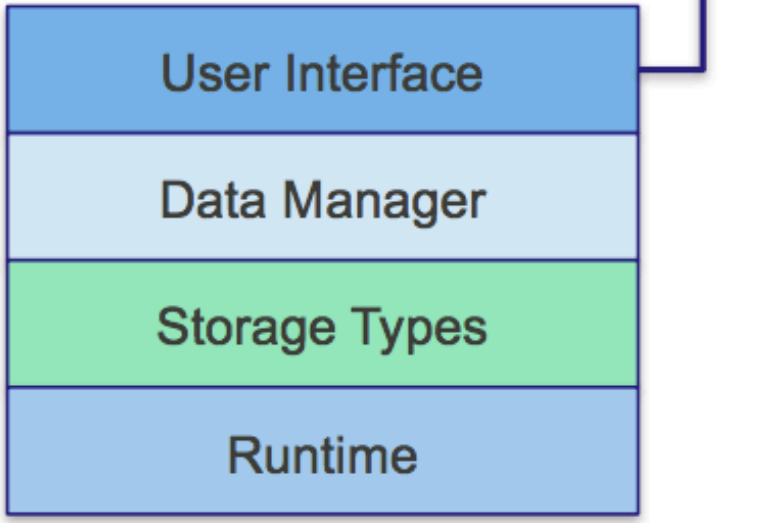


рахмат
danke
謝謝
ngiyabonga
Баярлалаа
spas
teşekkür ederim
спасибо
mersi
kia ora
barka
welalin
tack
mahalo
tapadh leat
faafetai lava
vinaka
blagodaram
dank je
misaotra
paldies
grazzi
спасибі
matondo
gracias
хвала
nanni
nandri
kiitos
dankie
dhanyavad
mauruuru
köszönöm
thank you
asante
manana
bedankt
bayarlalaa
gràcie
hvala
aciu
dankon
akun
obrigada
tenki
enkosi
djiere dieuf
tau
mochchakkeram
murakoze
дякую
mamnun
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
go raibh maith agat
chokrane
sobodi
dēkuji
obrigado
sagolun
sukriya
kop khun krap
taiku
arigatô
takk
dakujem
trugarez
mèsi
didi madloba
ありがとう
najis tuke
tanemirt
grazie
shukriya
mercé
мерси
kam sah hamnida
rahmat
terima kasih
rahmet
ευχαριστώ
diolch
dhanyavadagalu
তোমাকে ধন্যবাদ
감사합니다
xièxie
merci

# Back Up

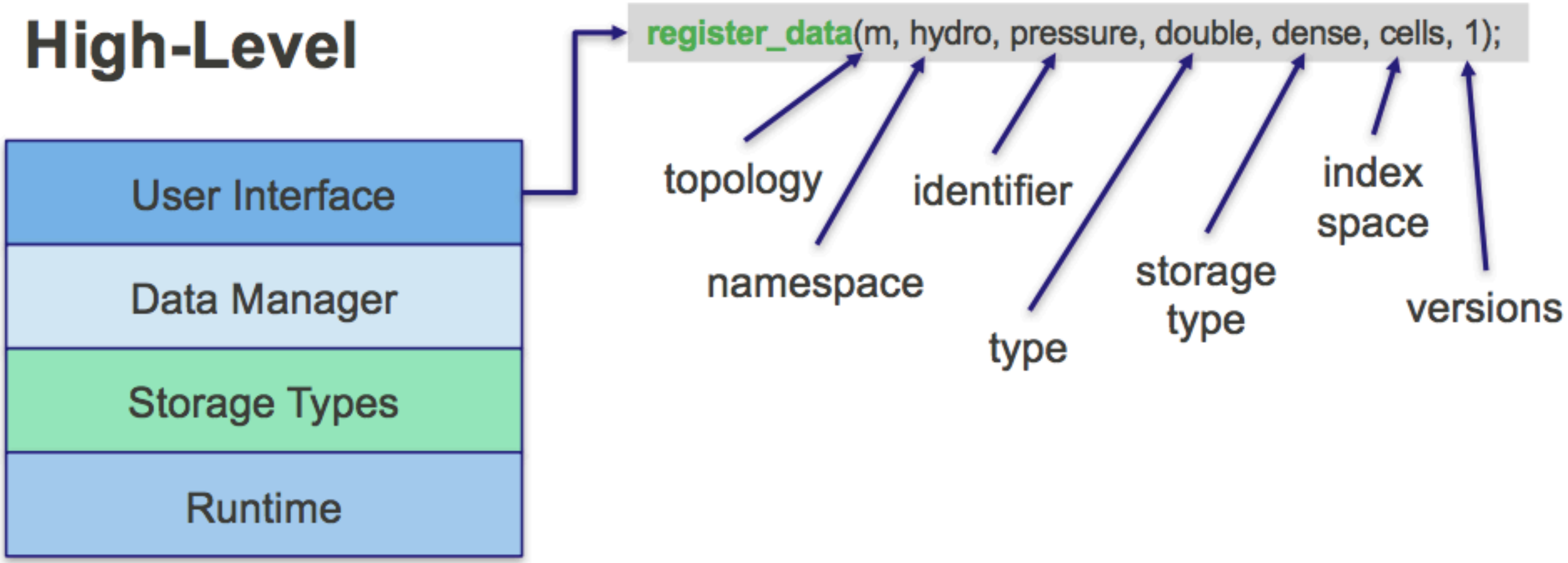
# FleCSI : Data Interface?

## High-Level



```
register_data(m, hydro, pressure, double, dense, cells, 1);
```

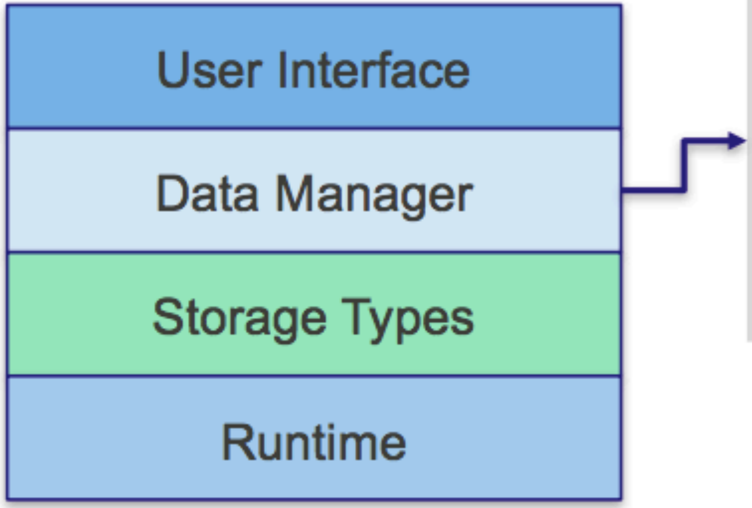
# FleCSI : Data Interface?



```
register_data(m, hydro, pressure, double, dense, cells, 1);
```

# FleCSI : Data Interface?

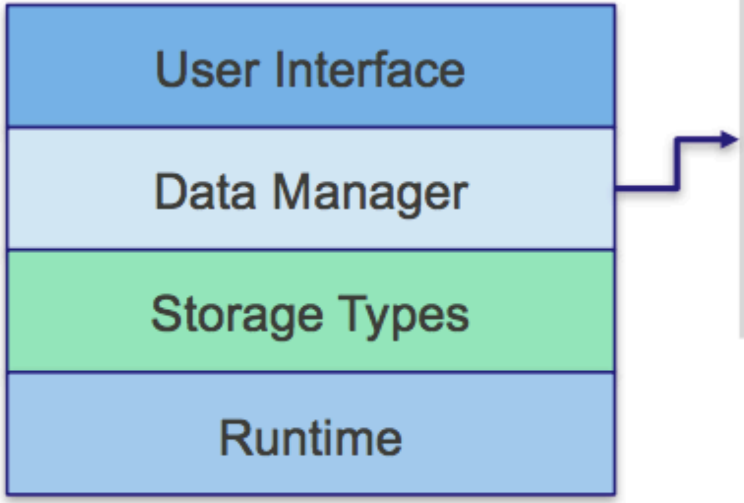
## Direct Interface



```
template<size_t ST, typename T, size_t NS,
typename ... Args>
decltype(auto) register_data(
data_client_t & data_client,
const const_string_t & key,
size_t versions=1,
Args && ... args)
```

# FleCSI : Data Interface?

## Direct Interface



```
template<size_t ST, typename T, size_t NS,
typename ... Args>
decltype(auto) register_data(
data_client_t & data_client,
const const_string_t & key,
size_t versions=1,
Args && ... args)
```

storage type

type

namespace

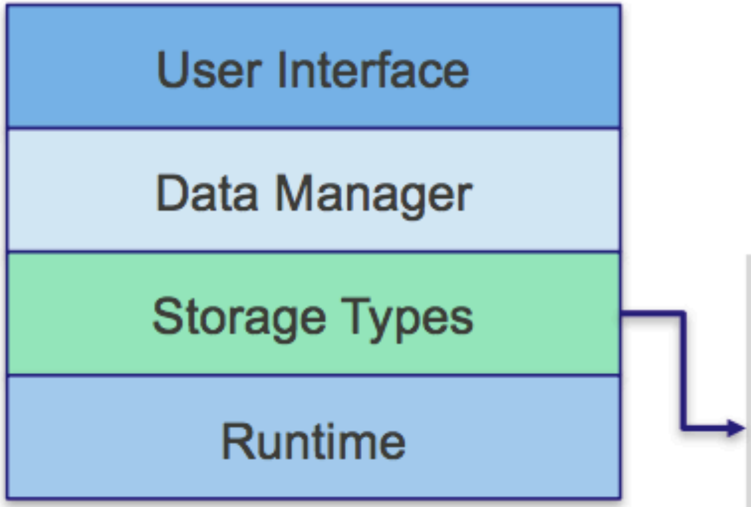
topology

identifier

versions

# FleCSI : Data Interface?

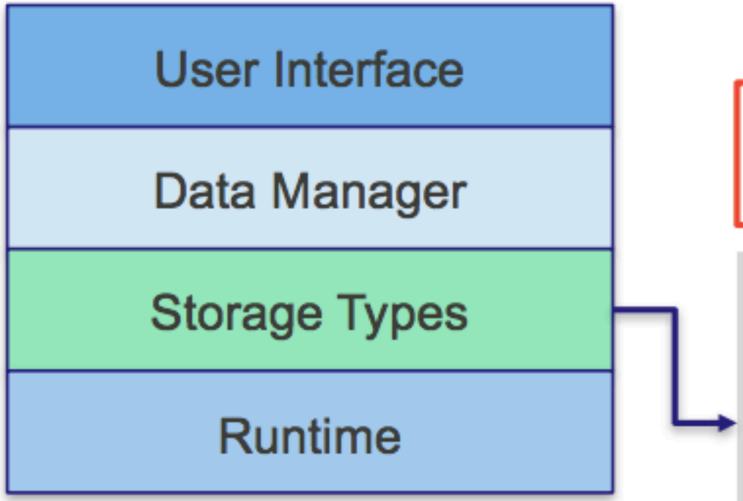
## Low-Level



```
template<typename T, size_t NS, typename ... Args>
static handle_t<T> register_data(
   data_client_t & data_client,
data_store_t & data_store, const const_string_t &
key, size_t versions, size_t index_space, Args && ...
args)
```

# FleCSI : Data Interface?

## Low-Level

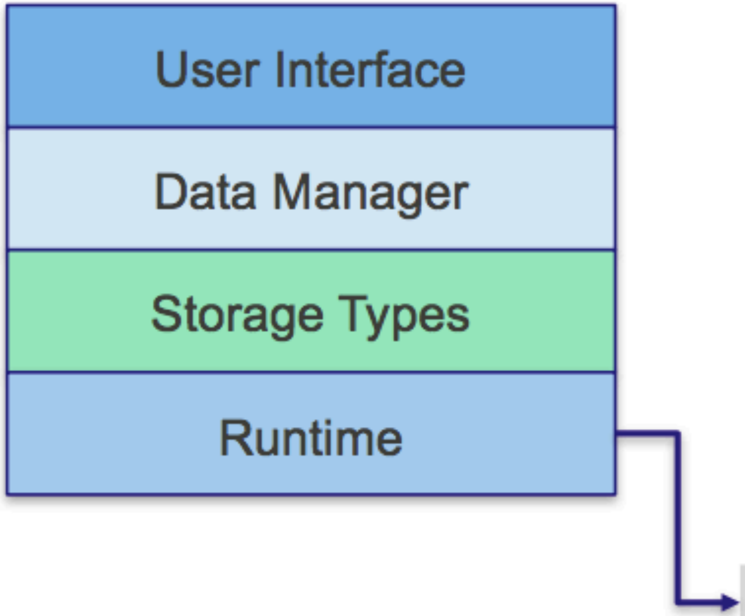


Mapped to specific storage type

```
template<typename T, size_t NS, typename ... Args>
static handle_t<T> register_data(
   data_client_t & data_client,
data_store_t & data_store, const const_string_t &
key, size_t versions, size_t index_space, Args && ...
args)
```

# FleCSI : Data Interface?

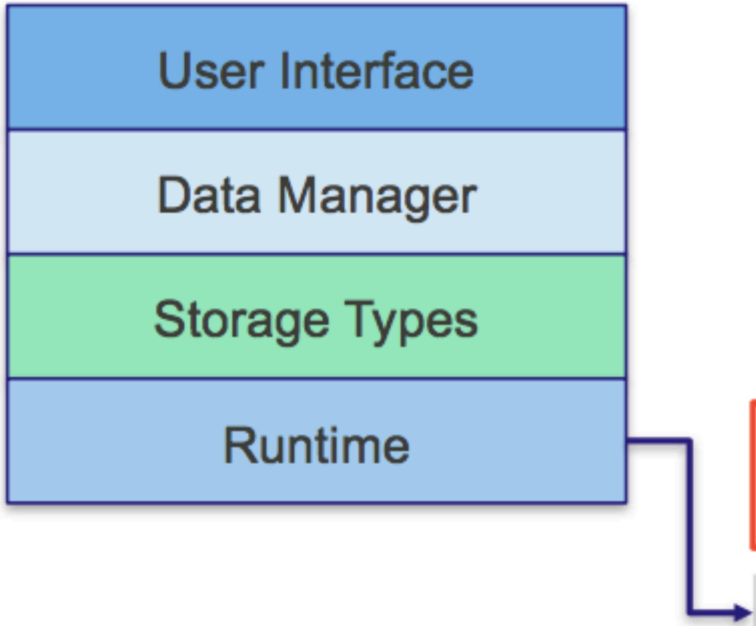
## Backend



```
FieldSpace fs = runtime->create_field_space(ctx);
```

# FleCSI : Data Interface?

## Backend



Storage type uses Legion runtime to create appropriate field space(s)

```
FieldSpace fs = runtime->create_field_space(ctx);
```